重排序列

2019年4月2日 19:12

一. 排序

1. 交换相邻两数使序列有序的最少交换次数(minimum swaps)=逆序数
2. 交换任意两数使序列有序的最少交换次数
   i. 即总数据量-循环节数量
   ii. 循环节指几个数依次占了下一个数应该出现的序号，形成环
   iii.
```
int x[10005];
bool v[10005];
map<int,int>m;
int n,t;
scanf("%d",&n);
for_(i,0,n){
    scanf("%d",x+i);
    m[x[i]]=i;}                    //记录初始下标
sort(x,x+n);                       //有序化
int cnt=0;                         //循环节数量
for_(i,0,n){
    if(!v[i]&& i!=m[x[i]]){        //找下标i的循环节
        v[i]=1;                    //注意：有些题即使i==m[x[i]]也要计数
        ++cnt;
        //类似bfs地找循环节所有元素
        for(int t=m[x[i]]; t!=i; t=m[x[t]])
            v[t]=1;}}
```
3. 交换任意两数使序列有序的最小代价（代价为两数的和）
   i. 若循环节内最小数mq足够小，让mq依次和节内其他n-1个数交换，即代价为mq*(n-1)+s(其他n-1个数)
   ii. 若mq不足够小，可以让整个序列最小的数mx和它交换，形成n+1个数的新循环节，再套上面的公式，即mx+mq+mx*n+s(所有n个数)
   iii. 具体足不足够小，交给stl来暴力算吧……
   iv.
```
int x[10005];
bool v[10005];
memset(v,0,sizeof(v));
map<int,int>m;
ull ans=0;                         //最小代价
vector<int>q;
int n,t;
scanf("%d",&n);
for_(i,0,n){
    scanf("%d",x+i);
    m[x[i]]=i;}                    //记录初始下标
sort(x,x+n);                       //有序化
int mx=x[0];                       //整个序列最小的数
for_(i,0,n){
    if(!v[i]&& i!=m[x[i]]){        //找下标i的循环节
```

```
            int tot=1;                  //当前循环节内数据量
            int mq=0x3f3f3f3f;          //当前循环节最小数
            v[i]=1;
            q.push_back(x[i]);
            mq=min(mq,x[i]);
            //给循环节所有元素入队
            for(int t=m[x[i]]; t!=i; t=m[x[t]]){
                v[t]=1;
                ++tot;
                q.push_back(x[t]);
                mq=min(mq,x[t]);}
            ans+=min(       //用mq作中介和用mx做中介
                accumulate(q.begin()+1,q.begin()+tot,(tot-1)*mq),
                accumulate(q.begin(),q.begin()+tot,(tot+1)*mx)+mq);
            q.clear();}}
    cout<<ans;
```

4. 任意交换次数例：换任一序列中任意两元素的位置，使$a_0b_0+a_1b_1+\cdots+a_{n-1}b_{n-1}$值最小，的最少交换次数
   i. 答案显然是序列长-循环节数量，问题是如何找循环节
   ii. 将a升序化，b降序化，此时ai和bi的对应关系即目标（a降b升也行）
   iii. 用a0，b0存有序化前的序列，ma0，mb0存有序化前数字对应的下标
   iv. 用ma[i]存a[i]对应的b[i]的原下标
   v.
```
const int Mn=100005;
int a0[Mn],b0[Mn];
bool va[Mn],vb[Mn];
map<int,int>ma0,mb0,ma,mb;
ull loop,loop2;

int main(){
    int n,q;
    scanf("%d",&n);
    for_(i,0,n){                //记录a初始下标
        scanf("%d",a0+i);
        ma0[a0[i]]=i;}
    for_(i,0,n){                //记录b初始下标
        scanf("%d",b0+i);
        mb0[b0[i]]=i;}
    vector<int> a(a0,a0+n);
    vector<int> b(b0,b0+n);
    sort(a.begin(),a.end(),less<int>());        //升序化
    sort(b.begin(),b.end(),greater<int>());     //降序化
    for_(i,0,n)                 //记录目标下标
        ma[a[i]]=mb0[b[i]],
        mb[b[i]]=ma0[a[i]];
    for_(i,0,n){
        if(!va[i]){         //找序列a中下标i的循环节
            va[i]=1;
            ++loop;
            for(int t=mb[b0[i]]; t!=i; t=mb[b0[t]]){
                va[t]=1;}}}
```

```
for_(i,0,n){
        if(!vb[i]){          //找序列b中下标i的循环节
                vb[i]=1;
                ++loop2;
                for(int t=ma[a0[i]]; t!=i; t=ma[a0[t]]){
                        vb[t]=1;}}}
cout<<min(n-loop,n-loop2);
return 0;}          //似乎loop恒等于loop2？直接交n-loop也能过
```